**Laboratory 2: Resampling - Image Rotation**

# **QUESTION 1**: On line 7, we use the inverse of the matrix T, why?

Every shear matrix has an inverse, and the inverse is simply a shear matrix with the shear element negated, representing a shear transformation in the opposite direction. Hence here, the inverse is used to shear the image to the left instead of the right.

# **QUESTION 2:** Change the value in T to a different value, i.e. −1. Around which point is the image sheared?

The image is sheared around the left of the image.

# **QUESTION** **3**: How can you change line 7 in the code so that the image is sheared around the center of the image (cx, cy) = (cols/2, rows/2) instead?

The image can be sheared around the center of the image by changing line 7 to: *([xg; yg] - [cols/2; rows/2]) + [cols/2; rows/2]*

# **QUESTION 4**: Compare the computational speed of the two codes for shearing, i.e. perform *shearimageFast*(im, T) and *shearimage*(im, T). Did you notice any time difference between the two? Tip: Use the commands *tic* and *toc*.

Both running time were very short, so we could not really notice the difference, but using the *Tic* and *Toc* commands we observed:

* With shearimage: Elapsed time is 0.177751 seconds.
* With shearimageFast: *Elapsed time is 0.108526 seconds.*

# **QUESTION 5**: Write an equation, similar to (1), for the affine transformation equation for rotation an angle θ around the point (xt, yt). Rotation of an angle:

# **QUESTION 6**: Write an alternative equation to (2) including the MATLAB command *round* instead of the sum and the rectangular function Π.

Alternative equative to (2):

# **QUESTION 7:** How does the difference-image look like?

The difference-image looks like this:

# How large and where are the errors?

We can see that the errors are rather small and are located around the writings of the logo.

# **QUESTION 8:** What is the error energy in the spatial domain when using nearest neighbor interpolation?

The error energy in the spatial domain is 134.

# **QUESTION 9:** Do the Fourier transform of the original and the forward and back-rotated image look similar?

While we can see some differences, they look very similar.

# **QUESTION 10:** Also, look at the Fourier transform of the difference image (*nIm-Im*). How are the errors distributed in relation to the frequencies?

The errors are very evenly distributed and sparse in relation to the frequencies.

# **QUESTION 11**: What is the error energy in the Fourier domain when using nearest neighbor interpolation?

The error energy in the Fourier domain is 134, exactly like in the spatial domain.

# **QUESTION 12**: Compare the error energy in the spatial and the Fourier domain. Which theorem relates these two measures and do your own measurements agree with the theorem?

It is the Perceval’s theorem.

# **QUESTION 13:** Do your own measurements agree with the theorem?

Our own measurements agree with the theorem since they are exactly the same.

# **QUESTION 14:** Why can this be a better measure?

The output of the relative error for the Fourier is 118. It can be a better measure because the error in high and low frequencies are normalized. Hence the difference in error (which are larger in high frequencies) don’t penalize the final result.

# Remember that in most images, the low frequencies have **higher** amplitude and the high frequencies have **lower** amplitude.

# **QUESTION 15:** Compute and display the relative error image for the Fourier components. How are the errors distributed in relation to the frequencies? A contrast interval between 0 and 2 is recommended here, i.e. use [0 2] as a parameter to *imagesc*.

As shown by the white dots, the errors are more present in the higher frequencies.

# **QUESTION 16**: The procedure for calculating bilinear interpolation was given in Lecture 4. Which slide(s) are you going to use to support your implementation of bilinear interpolation?

We will use the slides 17-20 of Lecture 4, while the 17 will be the most helpful.

# **QUESTION 17:** To check how the interpolation in the rotation affects the image, rotate and back-rotate the image as you did in section 3.1b. How does the difference-image look like? How large and where are the errors?

There seems to be less error than with the nearest neighbor interpolation. There are not many errors, they are mainly located on the edges of the image.

# **QUESTION 18:** What is the error energy in the spatial domain when using bilinear interpolation?

The error energy is in the spatial domain when using bilinear interpolation.

# Compare it with the result for nearest neighbor interpolation

This result is slightly lower than the nearest neighbor interpolation: 129 < 134, hence the bilinear interpolation seems better.

# **QUESTION 19:** Why are the high frequencies more attenuated in the forward and back-rotated image?

The edges are smoother so the high frequencies are more attenuated.

# **QUESTION 20:** Compute and display the relative error image for the Fourier components. Compare it with the previous one when you used nearest neighbor interpolation. How are the errors distributed in relation to the frequencies? A contrast interval between 0 and 2 is recommended here, i.e. [0 2].

Compare to the nearest neighbor interpolation, the relative error image is relatively the same, the errors seem to be evenly distributed across the frequencies.

# **QUESTION 21**: What is the error energy in the spatial domain when using bicubic4 interpolation? Compare it with the result for bilinear and nearest neighbor interpolation.

Our error energy when using bicubic4 is 94 which is better than using bilinear (129) and nearest neighbor (134).

# **QUESTION 22**: Compute and display the relative error image for the Fourier components. Compare it with the previous one when you used nearest neighbor interpolation and bilinear interpolation. How are the errors distributed in relation to the frequencies? A contrast interval between 0 and 2 is recommended here, i.e. [0 2].

The errors are distributed on the edges of the image as the others interpolations but there are less errors than before.

# **QUESTION 23**: Why do you get values below 0 and values above 1? (To get a nicer look you can supply the range [0 1]) to imagesc.)

We get values below 0 and above 1 because if we look at the equation (4) of we can see that depending on the value of we can have those values.

# **QUESTION 24**: What is the error energy in the spatial domain when using bicubic16 interpolation? Compare it with the result for bicubic4, bilinear and nearest neighbor interpolation

We found an error energy of 52 which is far less than bibubic4 (94), bilinear (129) and nearest neighbor (134).

# **QUESTION 25**: Compute and display the relative error image for the Fourier components. Compare it with the previous one when you used nearest neighbor interpolation and bilinear interpolation. How are the errors distributed in relation to the frequencies? A contrast interval between 0 and 2 is recommended here, i.e. [0 2].

The error seems to be distributed around the center but not in the center. Hence the errors are located around the lower frequencies rather than in the higher frequencies.

# **QUESTION 26:** Here, it should be clearly visible that bicubic16 interpolation gives the best image quality - right? Also show the images to the teacher! Is the teacher content?

Yes, bircubic16 interpolation clearly gives the best image quality.

# **QUESTION 27**: Execute the script and check the error energy in the spatial domain. Is it equal to what you got before for nearest neighbor interpolation in section 3.1?

Yes, it is equal, we get 134 just as we did for nearest neighbor interpolation.

# **QUESTION 28**: How can you change the script so that it performs bilinear interpolation instead?

We can do this by replacing “nearest” with “bilinear” is the function interp2.

# Do so and then check the error energy in the spatial domain. Is it equal to what you got before for bilinear interpolation in section 3.2?

The error energy is again equal, we get 129 as before.

# **QUESTION 29**: How can you change the script so that it performs bicubic16 interpolation instead?

We can do this by replacing “bilinear” with “bicubic16” is the function interp2.

# Do so and then check the error energy in the spatial domain. Is it equal to what you got before for bicubic16 interpolation in section 3.4?

Again it is equal, the error energy in the spatial domain obtained is 52.